

(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau



(43) International Publication Date
6 December 2001 (06.12.2001)

PCT

(10) International Publication Number
WO 01/93212 A2

- (51) International Patent Classification⁷: G07F 7/02, 19/00 (74) Agents: MALLIE, Michael, J. et al.; Blakely, Sokoloff, Taylor & Zafman LLP, 7th Floor, 12400 Wilshire Boulevard, Los Angeles, CA 90025 (US).
- (21) International Application Number: PCT/US01/17542
- (22) International Filing Date: 30 May 2001 (30.05.2001) (81) Designated States (*national*): AE, AG, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, BZ, CA, CH, CN, CO, CR, CU, CZ, DE, DK, DM, DZ, EE, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MA, MD, MG, MK, MN, MW, MX, MZ, NO, NZ, PL, PT, RO, RU, SD, SE, SG, SI, SK, SL, TJ, TM, TR, TT, TZ, UA, UG, US, UZ, VN, YU, ZA, ZW.
- (25) Filing Language: English
- (26) Publication Language: English
- (30) Priority Data: 09/583,815 30 May 2000 (30.05.2000) US (84) Designated States (*regional*): ARIPO patent (GH, GM, KE, LS, MW, MZ, SD, SL, SZ, TZ, UG, ZW), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE, TR), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, GW, ML, MR, NE, SN, TD, TG).
- (71) Applicant (*for all designated States except US*): POINTSEC MOBILE TECHNOLOGIES, INC. [US/US]; 1333 N. California Blvd., Suite 445, Walnut Creek, CA 94596 (US).
- (72) Inventors; and
- (75) Inventors/Applicants (*for US only*): MUIR, John, Richard [US/US]; 26 Julie Highlands Court, Lafayette, CA 94549 (US). LENNARTSSON, Kurt, Uno, Rudolf, Edgar [SE/US]; 1799 Candelero Court, Walnut Creek, CA 94598 (US). BILLSTROM, Leif, Olov [SE/SE]; Langsvevagen 32, S-865 33 Alno (SE).

Published:

— without international search report and to be republished upon receipt of that report

For two-letter codes and other abbreviations, refer to the "Guidance Notes on Codes and Abbreviations" appearing at the beginning of each regular issue of the PCT Gazette.

WO 01/93212 A2

(54) Title: APPARATUS AND METHODS FOR USING A VIRTUAL SMART CARD

(57) Abstract: An apparatus and method for providing a virtual smart card. The apparatus includes a smart card interface and a virtual smart card configured to emulate a physical smart card as if a smart card is connected with the smart card interface.

Printed by EAST

UserID: sahuja

Computer: TRN03985

Date: 05/07/2007

Time: 14:03

Document Listing

Document	Image pages	Text pages	Error pages
WO 200193212	33	0	0
Total	33	0	0

APPARATUS AND METHODS FOR USING A VIRTUAL SMART CARD

FIELD OF THE INVENTION

The present invention relates to computing systems and electronic devices. Specifically, the present invention relates to a virtual smart card for providing data security.

BACKGROUND

With the emergence of computing networks and portable electronic devices, sensitive data is commonly being exchanged and communicated. For example, a user on the Internet, or the World Wide Web (WWW) can exchange sensitive data to perform financial transactions such as the purchase of goods and services (commonly referred to as "e-commerce"). Furthermore, a user of portable electronic devices (e.g., electronic tokens, smart cards, personal data assistants, cellular phones, and other like devices) can also exchange and communicate such sensitive data. A problem with the exchange of sensitive data is security.

One type of system that has been developed to protect electronic data is a "cryptographic system." A cryptographic system converts plain, readable data into unreadable encrypted data ("ciphertext") using complex mathematical algorithms and an encryption key. To read ciphertext, a decryption process is required, which requires a decryption key. The decryption process converts the ciphertext into readable data using the decryption key along with complex mathematical algorithms.

One type of cryptography is "secret key" cryptography. Secret key cryptography requires identical encryption and decryption keys that are used to share sensitive data. The secret key is exchanged between users, and used to encrypt and decrypt data. A disadvantage of secret key cryptography is that the secret key has to be exchanged.

A more versatile form of cryptography is "public-key" (PK) cryptography. Public-key cryptography requires a key pair which consists of a public

encryption key ("public key") and a private decryption key ("private key"), which are unique for each user. In public-key cryptography, the public key is made available to the public such that any user may use the public key to encrypt data, which can only be decrypted by the private key of a receiving party. The public key is not used in the decryption process. As such, the public key and the private key do not have to be exchanged. The private key may also be used to sign data such that others can verify the originator of the data. For example, a user can create a digital signature with the user's private key to allow others to determine the identity of the user. A user's digital signature is specific data that can only be made by a user of a private key specific to the user.

Figure 1 illustrates a conventional PK security system 50 having a private key 45, which is accessible to a device operating system 25. For example, PK security system 100 may operate within the Microsoft® Windows® 2000 Public Key Infrastructure (PKI). The PKI provides a cryptographic application program interface (crypto API) 30. Crypto API 30 is an interface between the device operating system and cryptographic processes. Cryptographic processes are provided by cryptographic service providers. Typically, cryptographic service providers provide software or hardware functions that perform cryptographic algorithms to secure data for a computing system.

Referring to Figure 1, conventional PK security system 50 shows a computing system 10 including an application 20, device operating system 25, crypto API 30, and memory 40 storing private key 45. Computing system 10 is illustrated as an Internet access device running application 20. For purposes of explanation, application 20 is illustrated as a web browser application. For example, application 20 may be a Netscape® web browser or a Microsoft® Internet Explorer® web browser.

Application 20 operates with device operating system 25. For example, device operating system 25 is a Microsoft® Windows® family operating system. Application 20 is shown to perform a process that requires use of private key 45

stored in memory 40. For example, application 20 may allow a user to request a digital signature using private key 45 or decrypt data using private key 45.

For application 20 to use private key 45 stored in memory 40, application 20 must interact with device operating system 25. Device operating system 25 operates with crypto API 30 in providing data security processes. Crypto API 30 is an interface to private key 45 and performs cryptographic processes using private key 45. For example, crypto API 30 may perform cryptographic algorithms to generate a digital signature using private key 45. Thus, a user of application 20 can send the digital signature to another user connected in a networking environment with computing system 10.

A disadvantage of conventional PK security system 50 is that crypto API 30 is not isolated from the operating system. As a result, private key 45 is exposed to device operating system 25. By being exposed to device operating system 25, private key 45 may be exposed to unauthorized users who may have improperly accessed computing system 10 and device operating system 25. Furthermore, private key 45 may be exposed to computer viruses through device operating system 25 that may attack computing system 10 and destroy or alter private key 45.

Figure 2 illustrates another conventional PK security system 70 supporting a smart card interface 50 and a smart card 65. For example, conventional PK security system 70 may operate within a Microsoft® PC/SC smart card infrastructure. This infrastructure allows cryptographic service providers to provide smart cards that perform cryptographic functions. For example, a smart card may be used to provide a private key for a digital signature.

Referring to Figure 2, conventional PK security system 70 shows a computing system 10 including an application 20, device operating system 25, smart card interface 50, and smart card reader driver 55. For purposes of explanation, application 20 and device operating system 25 operate in a similar manner as in Figure 1. Smart card reader driver 55 is coupled with an external

smart card reader 60 that receives smart card 65. Smart card 65 stores a private key 67. Smart card 65 performs cryptographic functions using private key 67. Smart card 65 communicates with computing system 10 through smart card interface 50, smart card reader driver 55, and smart card reader 60.

Smart card 65 is a portable electronic device having embedded micro-circuitry that can store and process electronic data. Typically, smart card 65 is the size of a credit card having embedded memory to store data and an embedded processor to process data. Smart card 65 performs a number of functions. For example, smart card 65 may be used to verify a user's identity, permit access to a computer or a network, or purchase goods and services. Smart card 65 may also ensure both authentication and authorization of the user to perform specific functions using private key 67.

Smart card reader 60 is a device that interfaces smart card 65 with computing system 10. Computing system 10 and smart card reader 60 communicate with each other through smart card reader driver 55. Smart card reader driver 55 is computer code that is used by device operating system 25 to communicate with smart card reader 60. Device operating system 25 communicates with smart card reader 60 through smart card interface 50. Computing system 10 may be configured through smart card interface 50 to support various smart card interfaces, e.g., PKCS (Public Key Cryptographic Services) 11, Microsoft® PC/SC, Crypto API for smart cards.

Although a smart card can perform security processes internally, a disadvantage with system 70 is that a smart card reader and a smart card need to be purchased and managed. Furthermore, there is no standard interface for using smart cards, which causes incompatibility problems. The lack of convergence of smart card standards has effectively prevented manufacturers of computing systems and portable electronic devices from offering built-in smart card readers.

SUMMARY OF THE INVENTION

An apparatus includes a smart card interface and a virtual smart card. The virtual smart card is configured to emulate a physical smart card as if a physical smart card were connected with the smart card interface.

Other features and advantages of the present invention will be apparent from the accompanying drawings, and from the detailed description, which follows below.

BRIEF DESCRIPTION OF THE DRAWINGS

The present invention is illustrated by way of example and not limited in the figures of the accompanying drawings in which like references indicate similar elements and in which:

Figure 1 is an illustration of a conventional public-key security system having a private key accessible by an operating system;

Figure 2 is an illustration of another conventional public-key security system supporting a smart card;

Figure 3 is an illustration of an exemplary computer system supporting a virtual smart card;

Figure 4 is a block diagram of one embodiment of a computer system for operating a virtual smart card;

Figure 5 is an illustration of an exemplary networking environment for using virtual smart cards;

Figure 6 is an illustration of one embodiment of a memory space having a hidden memory space;

Figure 7 is an illustration of one embodiment of a virtual smart card driver logic;

Figure 8 is a flow diagram of one embodiment of a process for configuring a computer system for a virtual smart card;

Figure 9 is a flow diagram of one embodiment of a process for providing boot security for a computer system using a virtual smart card; and

Figure 10 is a flow diagram of one embodiment of a process for using a private key using a virtual smart card.

DETAILED DESCRIPTION

An apparatus including a smart card interface and a virtual smart card is described. The virtual smart card is configured to emulate a physical smart card.

In the following description, a virtual smart card is described that may perform calculations for security processes such as cryptographic calculations. The virtual smart card is not limited to any particular type of calculation. For example, the virtual smart card may perform a Rivest-Shamir-Adleman (RSA) calculation, Elliptic Curve Cryptography (ECC) calculation, a Digital Signature Algorithm (DSA) calculation, or another type of calculation. The virtual smart card may perform calculations for user authentication, access control, encryption, digital signatures, distributed authentication, secret-key agreement via public key, bulk data encryption, or other functions. Requests for a smart card are intercepted and processed by the virtual smart card.

An example of the function of the virtual card is as follows. A request to use a cryptographic function is received by the virtual smart card system. The request may be to use a private key. The private key is stored in a restricted memory space that is configured to be accessible by the virtual smart card. The request is validated by the virtual smart card. If the request is valid, the request is processed by the virtual smart card using the private key stored in the restricted memory space.

By using a virtual smart card that emulates and processes requests aimed at a physical smart card, the need to purchase and manage physical smart cards and smart card readers is obviated. Furthermore, the virtual smart card allows users to have physical smart card capability without the cost and compatibility issues of physical smart cards. The virtual smart card may be configured to support different types of smart card interfaces. Furthermore, the virtual smart card may further be configured to support dynamic change of virtual smart card types.

By having a private key stored in a restricted memory space, which is accessible by a virtual smart card, the private key is isolated from a device operating system. Because the private key is isolated from the device operating system, the private key is inaccessible to unauthorized users and computer viruses. By having a private key accessible to the virtual smart card, a standard way of providing data security designed for a smart card may be used.

Figure 3 is an illustration 100 of an exemplary computing system 110 supporting a virtual smart card 151. Computing system 110 includes virtual smart card 151 that emulates a smart card for computing system 110. Alternatively, virtual smart card 151 operates to perform access device interface (ADI) functions for computing system 110. For example, virtual smart card 151 may be used to authenticate a user prior to permitting access to the operating system of computing system 110.

In one embodiment, computing system 110 supports a PK infrastructure that operates with virtual smart card 151. Computing system 110 may operate with any smart card infrastructure. In one embodiment, the smart card infrastructure may be a Microsoft® PC/SC - CAPI smart card infrastructure, or a PKCS#11 infrastructure. Alternatively, computing system 110 may support other types of security infrastructures by adding the appropriate top level application programming interface (API), for example to replace PKCS#11.

Computing system 110 includes application 120, device operating system 125, smart card interface 150, and virtual smart card 151. Virtual smart card 151 includes virtual smart card driver 155 and memory 140. Memory 140 is a restricted memory space 170 used for storing private keys 167. Restricted memory space 170 and private key 167 are normally inaccessible to operating system 125 and applications. Virtual smart card 151 provides software code, which upon execution, emulates a physical smart card for device operating system 150.

Computing system 110 may be a general-purpose computer. Alternatively, computing system 110 may be a computing device such as a

personal data assistant (PDA), cellular phone, a laptop computer, workstation, or a server workstation. Computing system 110 operates with virtual smart card 151 believing virtual smart card 151 is a smart card.

For purposes of explanation, computing system 110 may operate as an Internet access device running application 120. Also, for purposes of explanation, application 120 is illustrated as a web browser application. For example, application 120 may be a Netscape® web browser or a Microsoft® Internet Explorer® web browser. Alternative applications 120 may be non-Internet related. Any application 120 may request certain smart card functions. Application 120 provides an interface between a user and computing system 110. Application 120 operates with device operating system 125 to perform functions for computing system 110.

Device operating system 125 is an interface between applications running on computing device 110, for example, application 120, and input and output devices coupled to computing device 110. Device operating system 125 may be a Microsoft® Windows® family operating system. Alternatively, device operating system 125 may be a Palm Pilot® operating system, or another operating system. In one embodiment, device operating system 125 sends commands designated for a smart card to smart card interface 150.

A user may use virtual smart card 151 to access computing system 110, as will be described below. A user may further encrypt and decrypt data using virtual smart card 151. For example, a user may request, through application 120, to encrypt data using private key 167.

Smart card interface 150 is coupled to virtual smart card 151 through virtual smart card driver 155. Virtual smart card driver 155 is software code, which upon execution, emulates a physical smart card. In one embodiment, virtual smart card driver 155 operates to intercept commands from device operating system 125 designated for smart card interface 150 and to process the commands. Virtual smart driver 155 may also perform cryptographic calculations and to send the calculated results to device operating system 125

such that device operating system 125 believes it is receiving results from a smart card.

Virtual smart card driver 155 may also provide software code, which upon execution, emulates smart card functions for security and financial transactions. Virtual smart card driver 155 may also provide software code, which upon execution, emulates a smart card in any number of formats such as a PKCS # 11 or a PC/SC - CAPI smart card format. In one embodiment, a user may select a desired smart card format to emulate. In one embodiment, the smart card format being emulated may be changed on the fly.

In one embodiment, virtual smart card driver 155 receives a request from a user through application 120. Virtual smart card 151 processes the request and sends a result to application 120. In one embodiment, virtual smart card 151 creates a restricted memory space 170 ("hidden memory space"). Space is designated as restricted memory space 170, making it generally inaccessible to the operating system and normal applications running on top of the operating system. For one embodiment, this may be done by altering pointers or marking the sector bad, to indicate to the operating system that the data in that portion of memory is inaccessible. For another embodiment, this may be done by encrypting the data in the space, thus making the data inaccessible to the operating system.

Virtual smart card driver 155 uses such restricted memory 170 to store private keys 167. Virtual smart card driver 155 may further use restricted memory space 170 to store encrypted and decrypted data. In one embodiment, the restricted memory space 170 is on the hard drive. In another embodiment, restricted memory space 170 may be any type storage media, i.e. random access memory (RAM), read-only memory (ROM), electrically erasable programmable ROM (EEPROM), flash memory, compact disc (CD), CDROM, floppy, etc.

Private key 167 is data that is unique to a specific user. Private key 167 may be used in a public-key (PK) infrastructure. Alternatively, private key 167 may be used in a secret key infrastructure. Private key 167 may be used for a

number of security functions such as providing digital signatures that are unique to private key(s) 167.

Restricted memory space 170 may include any number of keys with varying bit lengths. Private keys 167 are used by virtual smart card driver 155 to perform cryptographic calculations. For example, virtual smart card driver 155 may use private keys 167 to perform cryptographic calculations to encrypt or decrypt data or provide digital signatures. Restricted memory space 170 may also include private keys 167 for multiple users. Each user may have a separate private key 167, stored in a separate part of the restricted space.

Figure 4 is a block diagram of one embodiment of computer system 110 for operating virtual smart card 151. Computer system 110 includes processor 202 and digital signal processor 208 operatively connected with random access memory (RAM) 205 and read only memory (ROM) 204 through system bus 203. Processor 202 and digital signal processor 208 is also coupled to fixed disk 225, which may also be an optical disk drive or other storage mediums, through input/output bus 201. Alternatively, processor 202 and digital signal processor 208 may be coupled to multiple storage mediums through input/output bus 201. Processor 202 and digital signal processor 208 communicate data using system bus 203 and input/output bus 201.

System bus 203 and input/output bus 201 may also receive inputs from keypad 222, input device 223, and devices connected with PCMCIA interface 224. System bus 203 and input/output bus 201 may provide outputs to display 220, fixed disk 225, and output device 226 (such as, for example, an audio speaker). Memory and storage media 204, 205 may also include a flash memory, EEPROM, or any combination of the above.

Computer system 110 may be controlled by operating system software, which includes a file management system, such as, for example, a disk operating system, which is part of the operating system software. The file management system may be stored in non-volatile storage device 204 and may be configured

to cause processor 202 to execute the various functions required by the operating system to input and output data and to store data in RAM 205 and on ROM 204.

In one embodiment, virtual smart card driver 155 is software executed by processor 202 or digital signal processor 208 to perform functions for virtual smart card 151. In one embodiment, virtual smart card driver 155 is stored in fixed disk 225. Alternatively, virtual smart card driver 155 may be stored in RAM 205 or ROM 204. In alternate embodiments, virtual smart card driver 155 may be stored in other memory devices within computer system 110 such as an electrically erasable programmable memory (EEPROM).

Figure 5 is an illustration of an exemplary networking environment 250 for using virtual smart cards 151. Referring to Figure 5, networking environment 250 includes communication medium 255 coupling a plurality of computing devices with virtual smart cards 151. For example, a server 258, laptop computer 259, personal computer 257, and personal data assistant 256 each having a virtual smart card 151, may be coupled to communication medium 255. Alternatively, other types of devices having virtual smart card 151 may be coupled to communication medium 255. For example, an electronic token, cellular phone, and other electronic portable devices having a virtual smart card 151 may be coupled with communication medium 255.

Networking environment 250 may operate in any type of computer network. For example, the network may be a local area network (LAN), wide area network (WAN), or Internet network. Networking environment 250 may also support wired or wireless networking devices and environments. Networking environment 250 may include a plurality of networks. Furthermore, any number of devices may be interconnected with communication medium 255. Devices connected in networking environment 250 may include applications providing the ability to surf the Web, send e-mail, transfer files, or perform other functions. Such devices also include network protocol layers and device drivers to format data and to communicate the data on communication medium 255.

In one embodiment, computing devices coupled to communication medium 255 may include virtual smart card 151 emulating a smart card without using a smart card reader for a smart card. For example, a user of personal data assistant 256 may use virtual smart card 151 to log on to the networking environment 250. Furthermore, a user of personal data assistant 256 may send a user of laptop computer 259 a digital signature using private key 167 stored in virtual smart card 151 of personal data assistant 256. Similarly, other devices may use the virtual smart card 151 for similar functions.

Figure 6 is an illustration 300 of one embodiment of a memory space 305 including restricted memory space 170. Operating system 125, virtual smart card driver 155, and memory space 305 are shown. Memory space 305 may include restricted memory space 170. Alternatively, restricted memory space 170 and memory space 305 may be on separate memory devices.

In one embodiment, memory 305 is a fixed disk in which some "good" or "usable" sectors are marked as "bad." For example, virtual smart card driver 155 may alter a file allocation table (FAT) to mark a number of sectors "bad" on fixed disk memory 305 for restricted memory space. Similar methods may be used to mark sectors inaccessible for other types of file systems.

In one embodiment, memory 305 may be a ROM, RAM, flash memory or similar device, which is accessed using address pointers. For those types of memories, a portion of the memory may be designated as restricted memory space 170 by altering the address pointers used by the operating system to access the memory 305.

In one embodiment, restricted memory space 170 stores private key 167 (not shown). Because private key 167 is stored in restricted memory space 170, private key 167 is hidden from operating system 125. Restricted memory space 170 may store a plurality of private keys. Restricted memory space 170 may also store other types of data such as encrypted or decrypted data. In one embodiment, a plurality of separate restricted memory spaces may exist.

Virtual smart card driver 155 also includes an address pointer 171 to restricted memory space 170. In alternate embodiments, address pointer 171 may point to a number of addresses or locations for restricted memory space 170. In one embodiment, hidden memory space 170 is contiguous. Alternatively, hidden memory space 170 may be non-contiguous. In one embodiment, virtual smart card driver 155 uses address pointer 171 to send data to and retrieve data from restricted memory space 170.

Figure 7 is an illustration of one embodiment of virtual smart card logic 156. The following description of virtual smart card logic 156 represents software code functions and process block diagrams. In one embodiment, the functions and processes are performed using virtual smart card driver 155. Referring to Figure 7, virtual smart card logic 156 includes operating system interface 410, crypto logic 414, key purge mechanism 423, user interface 412, key generation unit 416, stored data 420, and private key 367.

Operating system interface 410 is an interface between operating system 125 and crypto logic 414. Operating system interface 410 receives data and requests from operating system 125. For example, a user of application 120 may request to access data on a smart card. Application 120 forwards the request to operating system 125, which looks for a smart card driver to communicate with a smart card. Operating system interface 410 receives the requests from application 120 via operating system 125. In one embodiment, operating system interface 410 identifies requests to smart card interface 150, and intercepts the request such that virtual smart card logic 156 processes the request being issued.

Crypto logic 414 operates to perform cryptographic calculations for virtual smart card. Crypto logic 414 performs cryptographic calculations using private key 422. Private key 422 may be stored encrypted in restricted memory space 367. For example, a user may send data and a request to encrypt the data. Operating system interface 410 receives the request and data, and forwards the request and data to crypto logic 414.

Crypto logic 414 processes the request by performing cryptographic calculations using private key 422. Crypto logic 414 forwards the result to the requesting application through operating system interface 410.

Key generation unit 416 operates to generate a symmetric key using a user password/ID from user interface 412. User interface 412 receives a user password/ID. In one embodiment, the user is asked to provide a user password/ID at login. In another embodiment, the user provides a password/ID whenever a cryptographic function is requested.

The symmetric key generated by key generation unit 416 may be used for encrypting and decrypting private key 367. In one embodiment, key generation unit 416 decrypts the encrypted private key 367, and forwards the unencrypted private key to crypto logic 414.

In one embodiment, crypto logic 414 stores the unencrypted private key from key generation unit 416 as private key 422, and performs calculations using private key 422. In one embodiment, after crypto logic 414 performs a calculation using private key 422, key purge mechanism 423 purges unencrypted private key 422 from crypto logic 414. Alternatively, unencrypted private key 422 may be purged after a period of time, during a shut down process, or when a user logs out of the system.

Figure 8 is a flow diagram of a process for installing a virtual smart card system on a computing system 110. For purposes of explanation, process 500 begins at processing block 505.

Referring to Figure 8, at processing block 505, virtual smart card software is installed. In one embodiment, virtual smart card software may be obtained via a network, or installed from a CD-ROM, or other medium. In another embodiment, virtual smart card 151 software may be pre-installed on computing system 110 or may operate from an external media, and the step of installing virtual smart card 151 software may be omitted.

At processing blocks 508 and 510, virtual smart card software designates restricted memory space within memory. Restricted memory space is memory

allocated in some way so that it is not directly accessible from the operating system.

In one embodiment, the restricted memory space is in sectors of the hard drive marked as "bad" sectors. In an alternate embodiment, virtual smart card software may alter pointers to designate memory such as an EEPROM as restricted memory space 170. Virtual smart card software accesses restricted memory space 170 by using an address pointer that indicates the location of restricted memory space 170.

At processing block 512, computing system 110 requests from a user a user password/ID. As discussed above, the user password/ID combination may be any type of unique user identification mechanism.

At processing block 514, a symmetric key is generated from the user password/ID. In one embodiment, user password/ID is padded to a known length, and this padded data is used to generate the symmetric key. Alternatively, the user password/ID may be used as-is to generate the symmetric key.

At processing block 516, key generation unit 416 decrypts an encrypted private key (uploaded with the virtual smart card software) using a key included with virtual smart card software. For example, private key may be "123," which encrypted may be "345." Thus, the key included with the virtual smart card software may be a digit - 2 function to decrypt "345" into "123."

At processing block 524, the now unencrypted private key is encrypted with the symmetric key generated at block 514. For example, private key "123" may be encrypted as "ABC." At processing block 525, the encrypted private key is stored in the restricted memory, for example stored on the disk. In one embodiment, the above two steps, e.g. decrypting the private key and encrypting the private key with the symmetric key may be performed in a different order, such that there is never a point at which the private key is unencrypted. The symmetric key and unencrypted private key are then purged

from the system. At this point, the only way of accessing the private key is be re-generating the symmetric key using the user password/ID.

At processing block 528, the user is asked whether boot up security should be provided by virtual smart card 151. If the user does not wish to use virtual smart card 151 for boot security, process 500 ends. If the user does wish to use the boot security provided by virtual smart card 151, at processing block 530 virtual smart card driver 155 alters the boot sector for computing system 110 to load the boot security provided by virtual smart card 151 each time computing system 110 is booted. Process 500 then ends.

Figure 9 is a flow diagram of one embodiment of a process 600 for providing boot security for computing system 110 using virtual smart card 151. For purposes of explanation, process 600 begins at processing block 605. The process starts during the computer boot-up process.

At processing block 605, a user is requested to input a user password/ID. User password/ID refers to any type of user identification, including PIN, Smart Card, Password Generator, biometric identification, etc. Computing system 110 receives the input user password/ID and forwards the user password/ID to virtual smart card logic 155.

At processing block 608, the user password/ID is padded to a known length. In one embodiment, this step may be skipped.

At processing block 610, a symmetric key is generated using the padded user password/ID.

At processing block 612, the encrypted private key is decrypted using the symmetric key generated at block 610.

At processing block 614, if the decryption process was unable to produce the correct private key, process 600 ends at processing block 616, and the computer does not fully boot up. The process may end at a later time, for example, prior to permitting access to the hard drive, but after completing the booting process. For example, if the user inputs an incorrect user password/ID, the decryption process will use an incorrect symmetric key and will produce an

incorrect private key. In one embodiment, the private key may be tested by decrypting some data encrypted with the user's public key.

At processing block 618, if the decryption process was able to produce the correct private key, the computing system 110 finishes booting and process 600 ends. At this point, the computer system is fully functional and may be used by the user. In one embodiment, the private key obtained at block 612 may be stored unencrypted, and used by the user, until the user logs of the system. Alternately, the private key generated at block 612 may be time limited, and the user may have to enter his or her user ID/password to use any cryptographic functions.

Figure 10 is a flow diagram of one embodiment of a process 700 for using virtual smart card. For purposes of explanation, computing system 110 is configured to use virtual smart card driver 155 and process 700 begins at processing block 705.

At processing block 705, a user sends a cryptographic request from application 120. This application may be any application, such as a browser, an e-mail client, document generating software, or any other application. For example, a user may request through application 120 to sign data using a smart card. Application 120 forwards the request to operating system 125, which looks for a device driver for smart card interface 150.

At processing block 708, virtual smart card driver 155 determines whether the request is for a smart card. If the request is not for a smart card, at processing block 710, virtual smart card driver 155 does not provide a response to operating system 125. The process then terminates, until a new request is received from the user.

At processing block 712, if the request is for a smart card (e.g., a request to use a smart card for a digital signature or other function), virtual smart card driver 155 intercepts the request. In one embodiment, the virtual smart card driver 155 provides the user with a user interface to input a user password/ID. In another embodiment, the user password/ID may already be in memory.

At processing block 714, the user password/ID is padded to a known length. In one embodiment, this step may be skipped.

At processing block 716, a symmetric key is generated from the padded password/ID.

At processing block 718, private key is decrypted using symmetric key generated at block 716.

At processing block 720, if the decryption process was unable to produce a correct private key, the process 700 continues to processing block 726. At processing block 726, the user is notified that an incorrect password/ID was received. In one embodiment, the user may be asked to re-enter a password/ID. Alternatively, the user is notified that no request was processed. Process 700 then ends.

If the decryption process was able to produce a correct private key, process 700 continues at processing block 722.

At processing block 722, the request is processed using the private key. For example, if a user requested to sign data using private key 422, crypto logic 414 performs a cryptographic calculation using private key 422 to generate a digital signature unique to private key 422, and thus to user. In another embodiment, crypto logic 414 may perform cryptographic calculations to encrypt data sent from application 120 using private key 422. At processing block 724, the result of the request, for example, signed data, is returned to the requesting user/application.

In the foregoing specification, the invention has been described with reference to specific exemplary embodiments thereof. It will, however, be evident that various modifications and changes may be made thereto without departing from the broader spirit and scope of the invention as set forth in the appended claims. The specification and drawings are, accordingly, to be regarded in an illustrative rather than a restrictive sense.

CLAIMS

What is claimed is:

1. An apparatus, comprising:
a smart card interface; and
a virtual smart card configured to emulate a physical smart card as if a physical smart card is connected with the smart card interface.
2. The apparatus of claim 1, wherein the virtual smart card is configured to emulate security functions of the physical smart card.
3. The apparatus of claim 2, wherein the virtual smart card is configured to perform functions using a private key that is accessible by the virtual smart card.
4. The apparatus of claim 3, wherein the private key is stored in a restricted memory space.
5. The apparatus of claim 4, wherein the virtual smart card includes:
a virtual smart card driver coupled with the smart card interface; and
a restricted memory storing the private key in the restricted memory space, the restricted memory is configured to be accessible by the virtual smart card driver.
6. The apparatus of claim 5, wherein the virtual smart card driver includes:
a crypto logic configured to perform cryptographic calculations using the private key.
7. The apparatus of claim 6, further comprising:
a key purge mechanism configured to purge the unencrypted private key used by the crypto logic.
8. The apparatus of claim 7, further comprising:
a key generation unit configured to generate a symmetric key and to encrypt and decrypt data using the symmetric key.
9. The apparatus of claim 8, wherein the data decrypted by the key generation unit includes the encrypted private key.

10. The apparatus of claim 8, wherein the crypto logic is configured to process a request for user authentication, access control, encryption, digital signatures, distributed authentication, secret-key agreement via public key, or bulk data encryption.
11. The apparatus of claim 5, wherein the virtual smart card driver is configured to intercept a smart card command designated for the smart card interface and to process the command.
12. A system comprising:
 - a virtual smart card;
 - a memory storing a private key, the private key stored so as to be only accessible by the virtual smart card; and
 - a processor configured to operate with an operating system and to perform functions for the virtual smart card such that operating system believes the virtual smart card is a smart card.
13. The system of claim 12, further comprising:
 - a smart card interface coupled with the virtual smart card, the virtual smart card configured to intercept a command to the smart card interface and to process the command.
14. The system of claim 12, wherein the virtual smart card performs security functions using the private key stored in the restricted memory space.
15. A method for configuring a computing system for a virtual smart card, the method comprising:
 - designating a restricted memory space, the restricted memory space being configured to be accessible by the virtual smart card; and
 - storing a private key in the restricted memory space to be used by the virtual smart card.
16. The method of claim 15, wherein designating a restricted memory space comprises:
 - making an area in memory inaccessible to an operating system and to applications running on top of the operating system.

17. The method of claim 15, further comprising:
 - requesting a user authorization;
 - generating a symmetric key with the user authorization; and
 - decrypting an encrypted private key as the private key stored in the restricted memory space.
18. The method of claim 15, further comprising:
 - altering a boot sector for the computing system to use the virtual smart card for a boot security process.
19. A method of using a virtual smart card to boot a computing system, the method comprising:
 - initiating a boot process for the computing system;
 - decrypting a private key by the virtual smart card using a symmetric key;
 - determining if the decrypted private key is valid; and
 - finishing the boot process if the decrypted key is valid.
20. The method of claim 19, wherein decrypting a private key includes:
 - requesting a user authorization; and
 - generating the symmetric key with the user authorization.
21. A method for using a virtual smart card, the method comprising:
 - receiving a request;
 - determining if the request is for a smart card; and
 - intercepting the request and processing the request by the virtual smart card if the request is determined to be for a smart card.
22. The method of claim 21, wherein processing the request processes the request using an encrypted private key stored in a restricted memory space.
23. The method of claim 22, further comprising:
 - decrypting the encrypted private key for processing the request; and
 - purging a decrypted private key after processing the request.
24. The method of claim 22, wherein processing the request includes:
 - validating the request; and
 - performing the request if the request is validated.

25. The method of claim 24, wherein validating the request includes:
requesting a user authorization;
generating a symmetric key with the user authorization;
decrypting a private key with the generated symmetric key; and
determining if the decrypted private key is valid.
26. The method of claim 21, wherein the request is for one of the following:
user authentication, access control, encryption, digital signatures, distributed authentication, secret-key agreement via public key, or bulk data encryption.
27. A method for using a virtual smart card, the method comprising:
receiving a request to a private key stored in a restricted memory space,
the restricted memory space being configured to be accessible to the virtual smart card;
validating the request by the virtual smart card;
if the request is valid, processing the request using the private key in the restricted memory space by the virtual smart card.

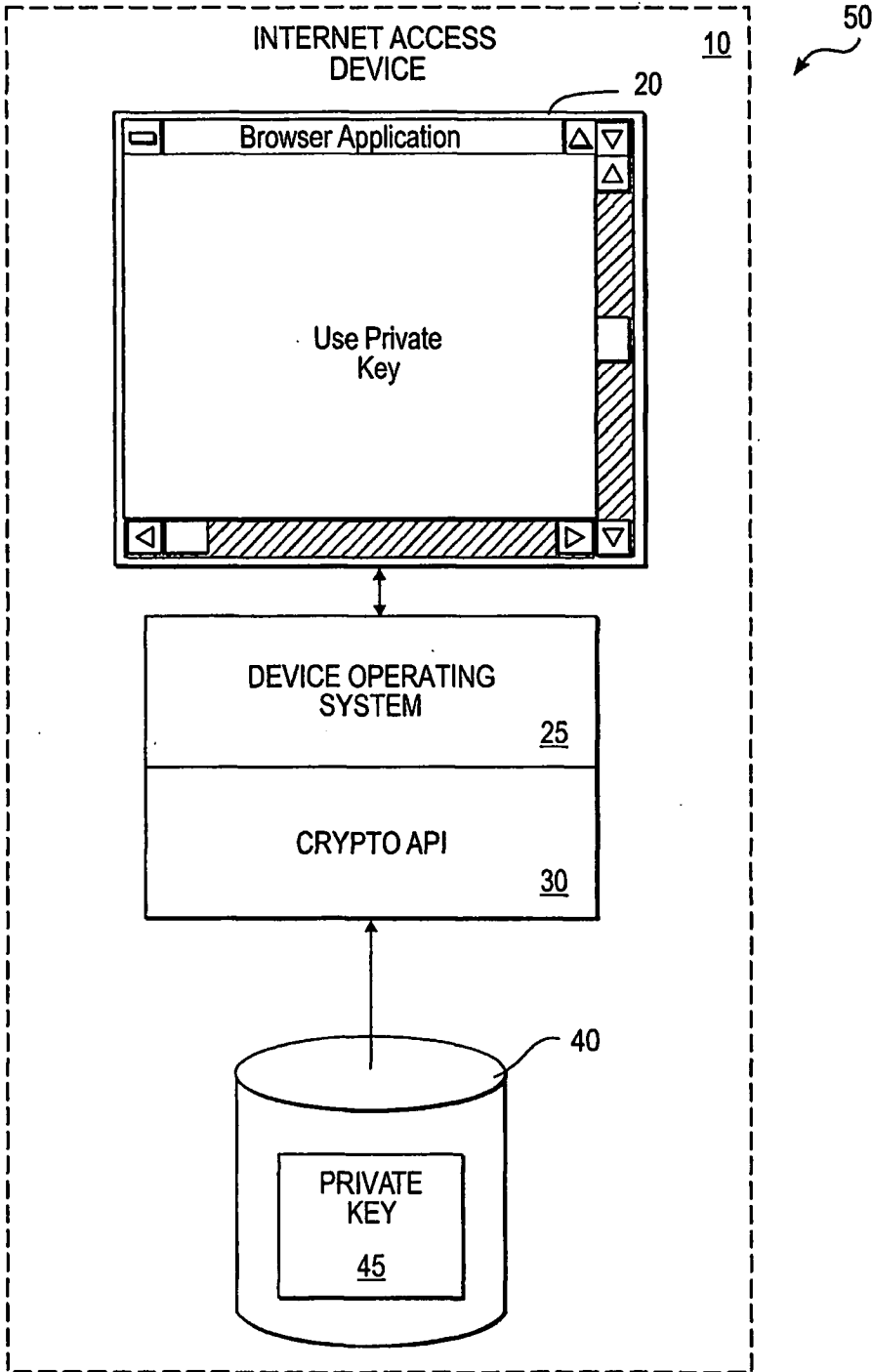


Fig. 1
(CONVENTIONAL ART)

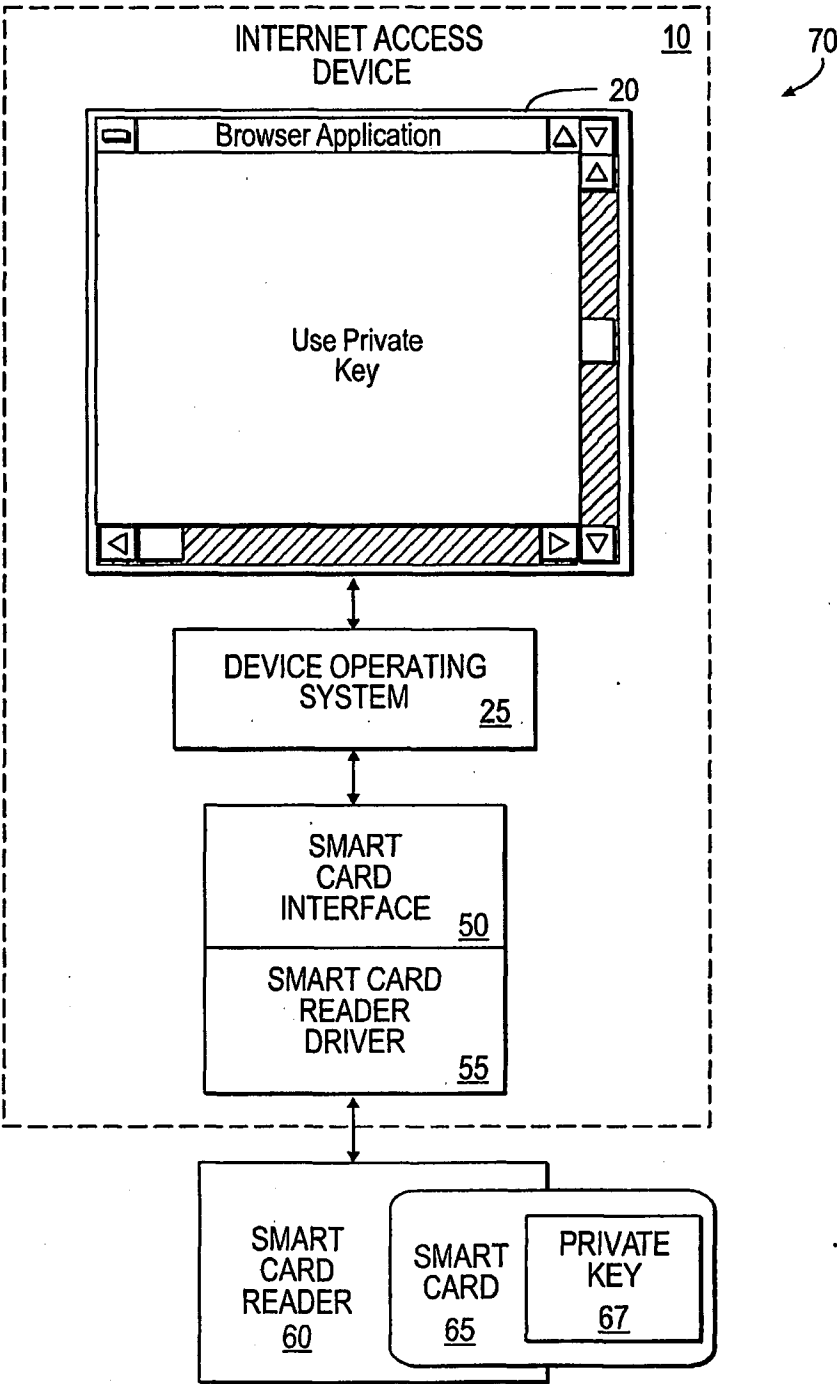


Fig. 2
(CONVENTIONAL ART)

3/10

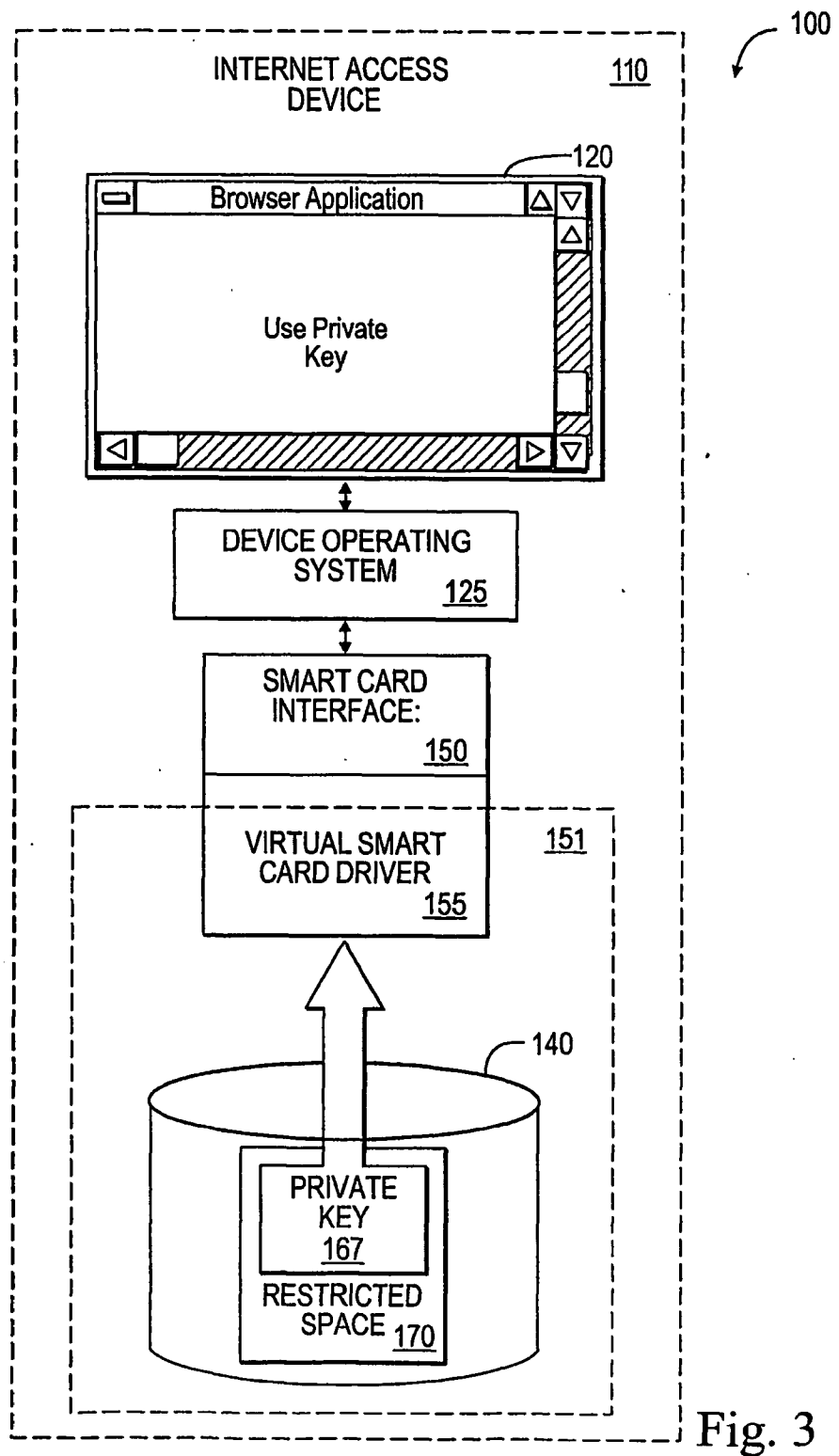


Fig. 3

4/10

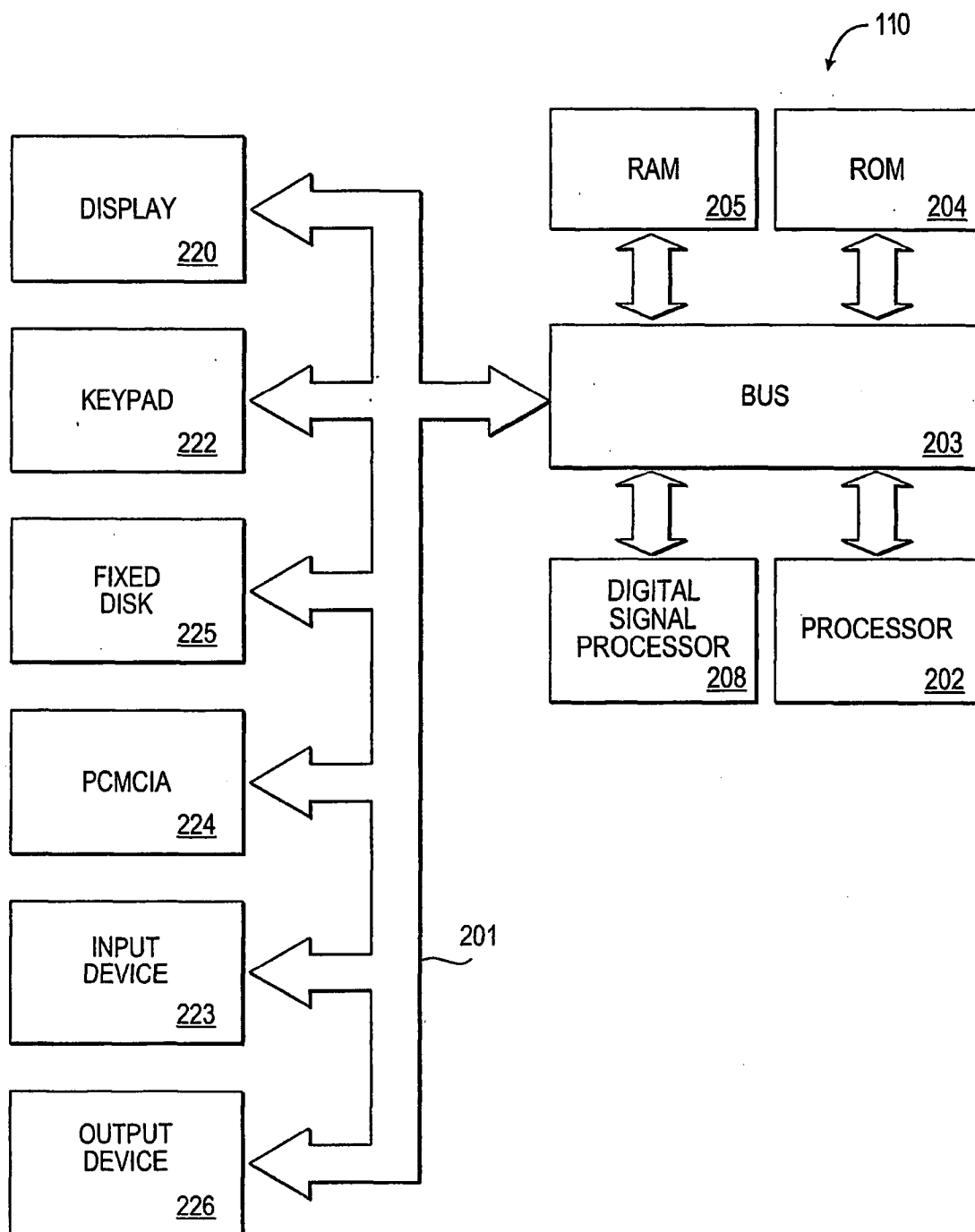


Fig. 4

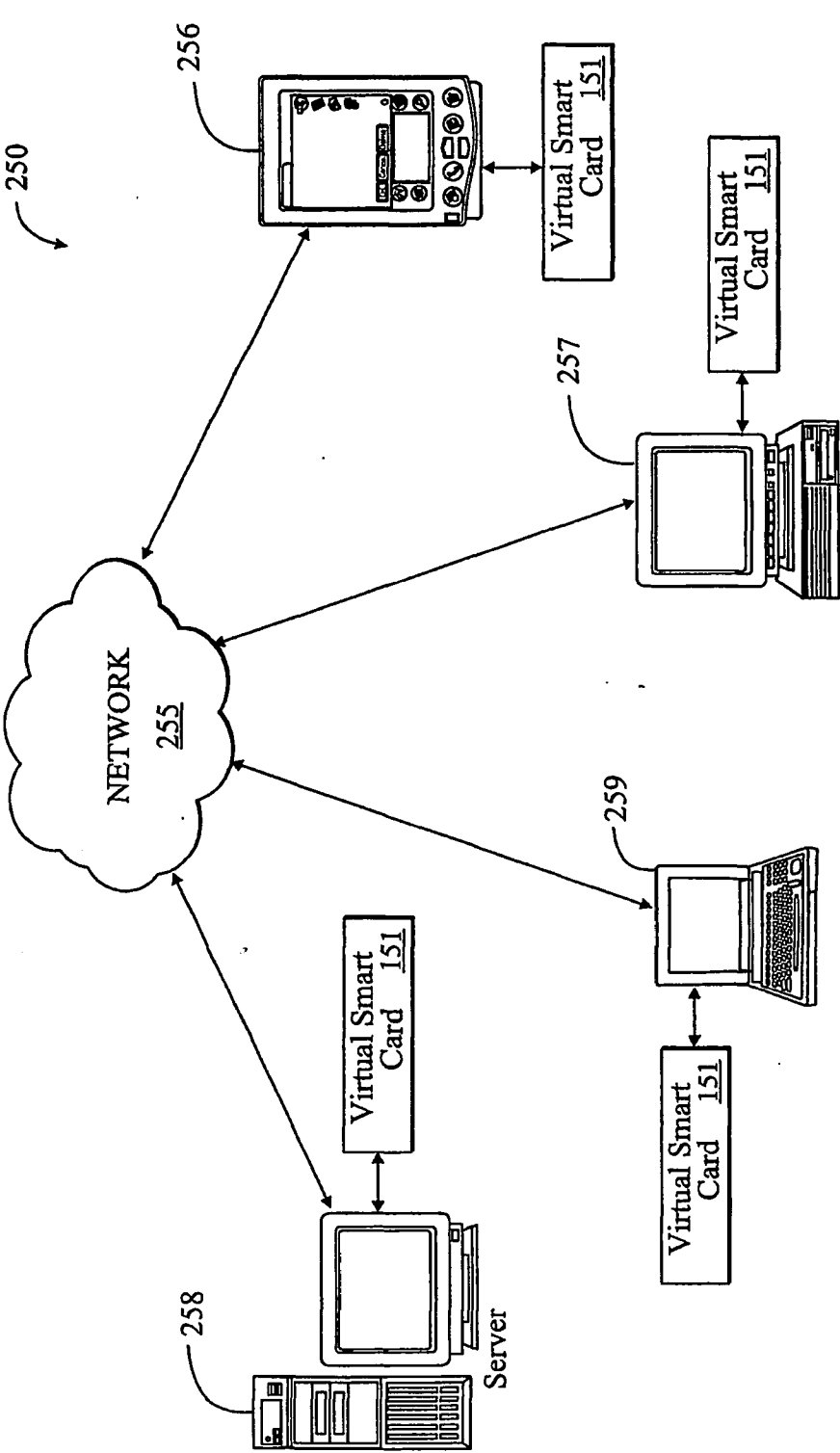


Fig. 5

6/10

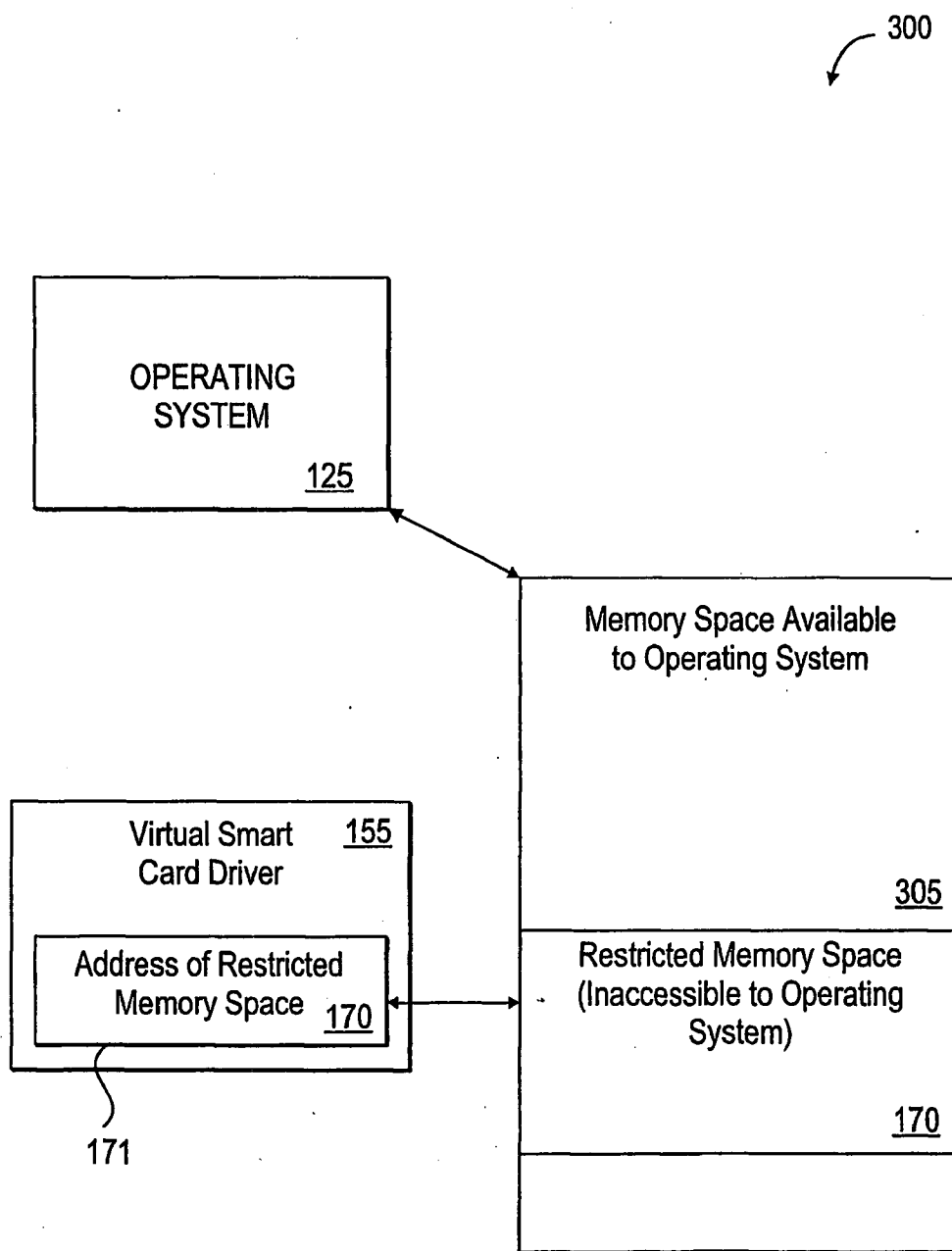


Fig. 6

7/10

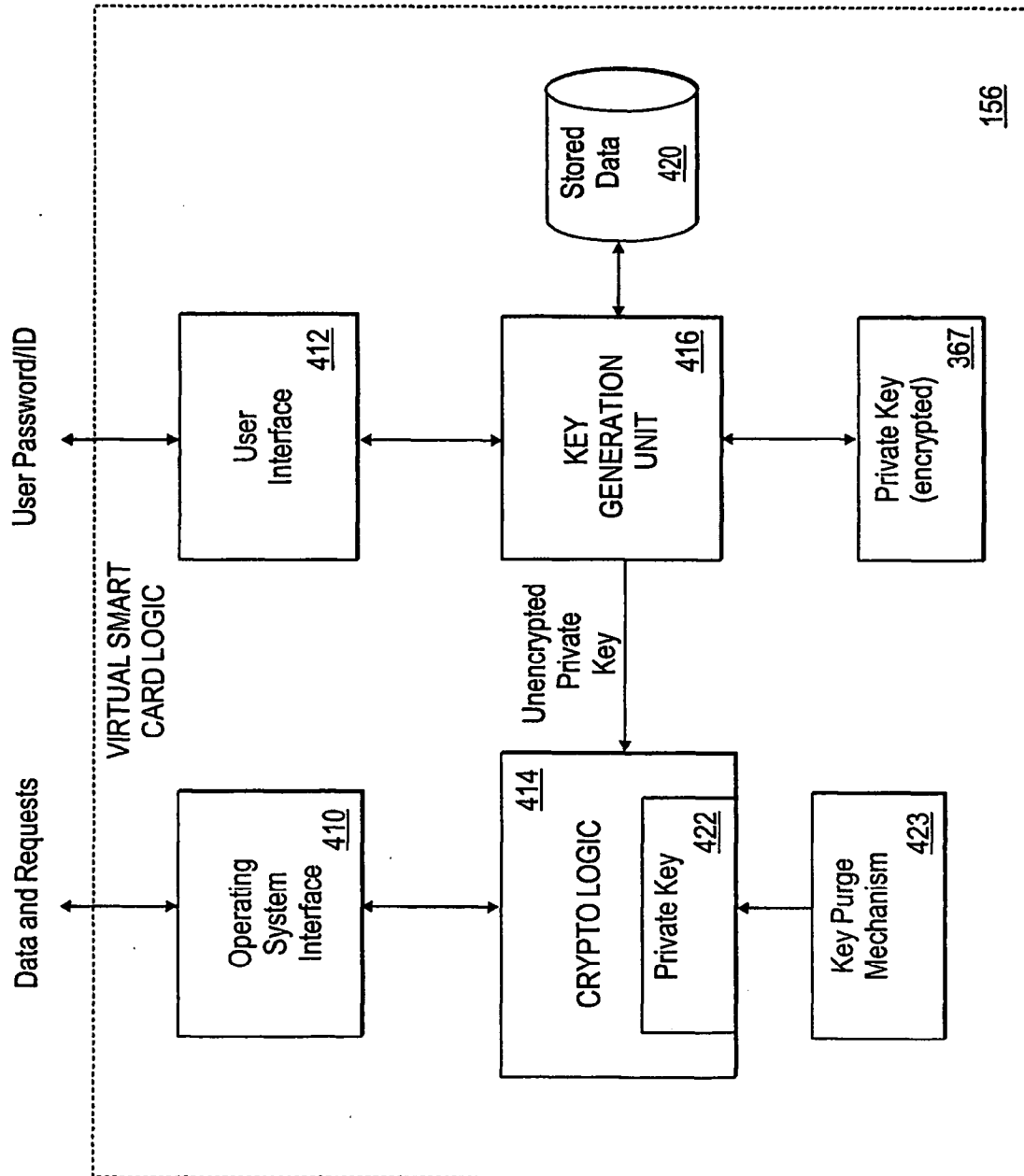


Fig. 7

8/10

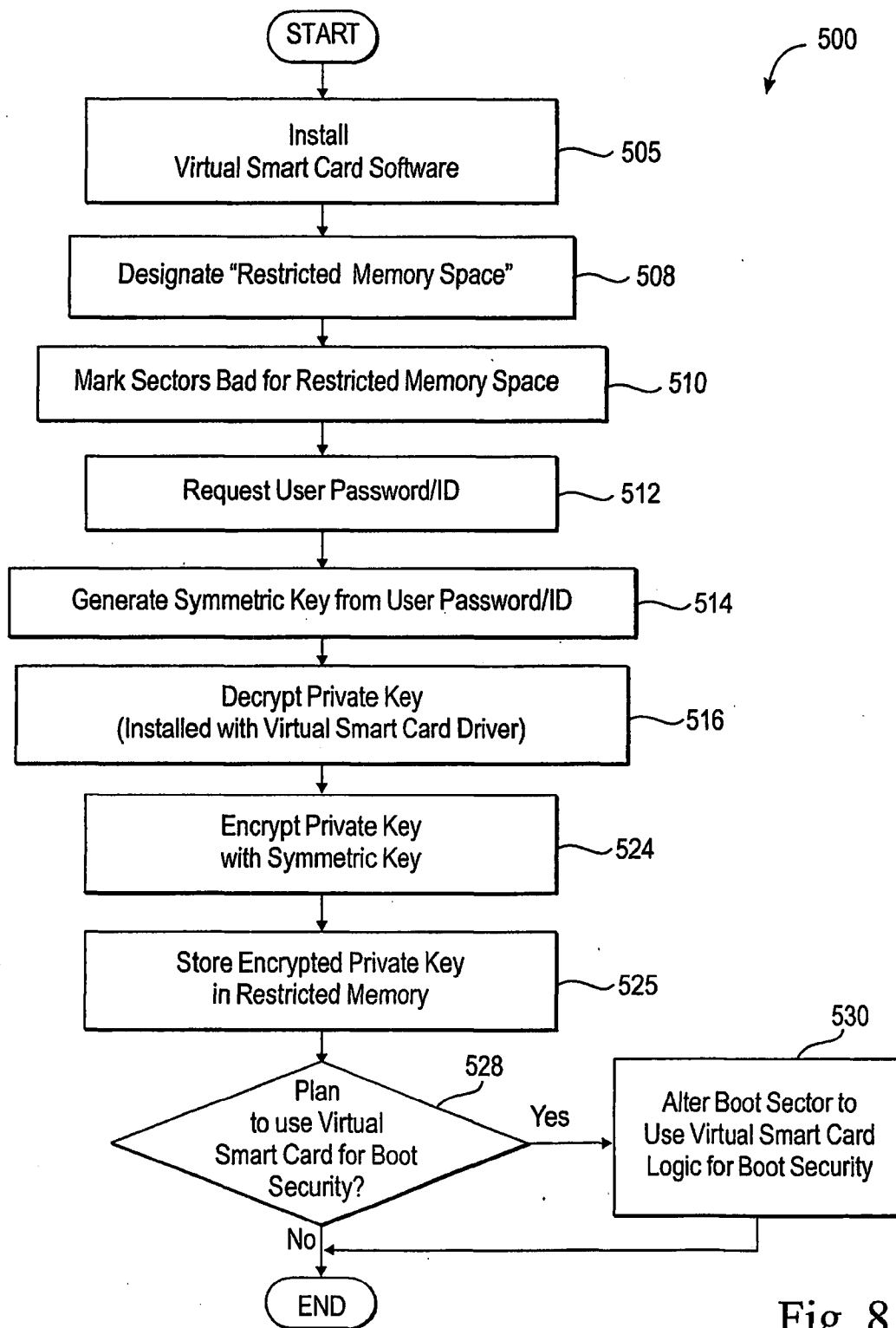


Fig. 8

9/10

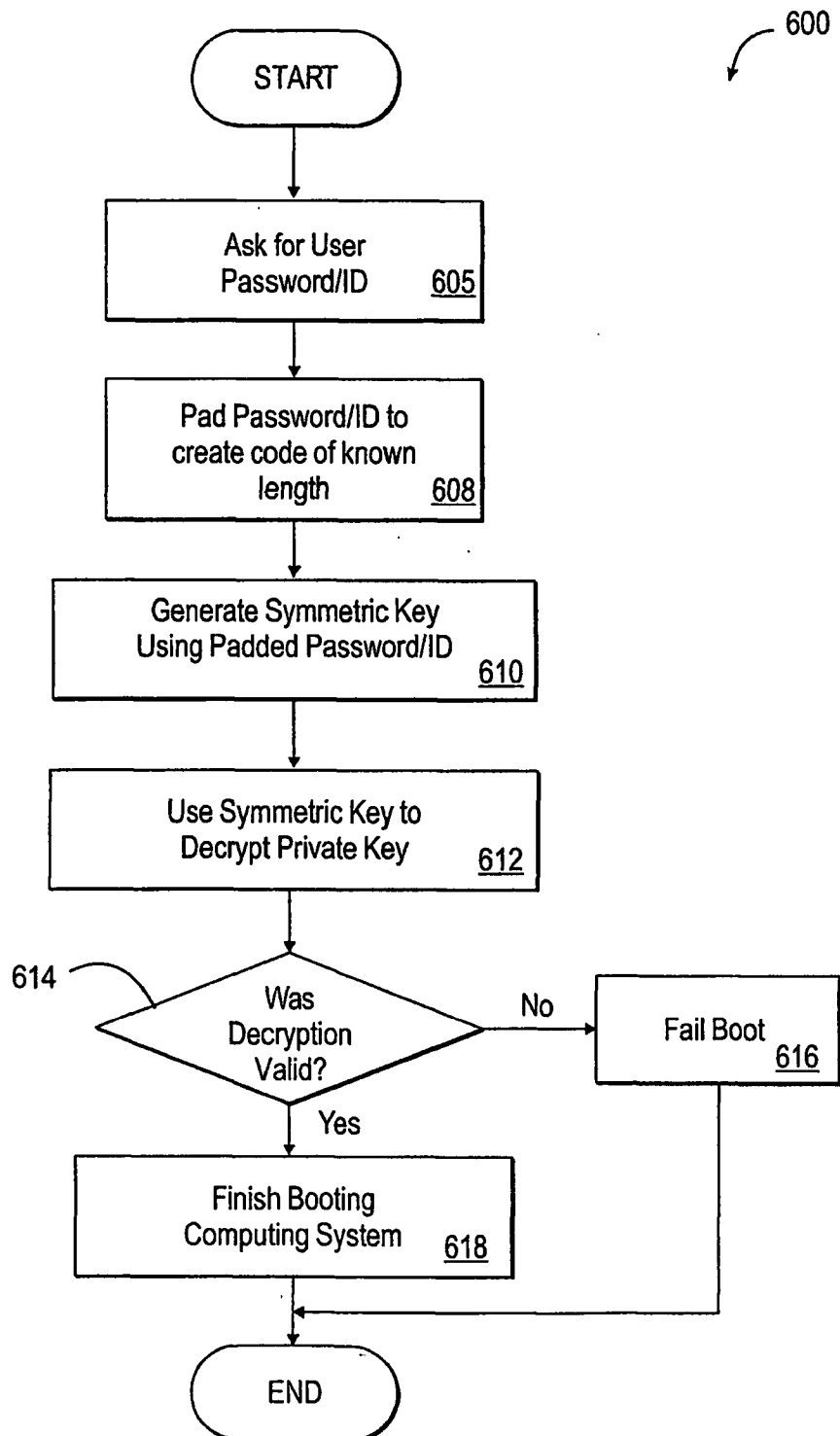


Fig. 9

10/10

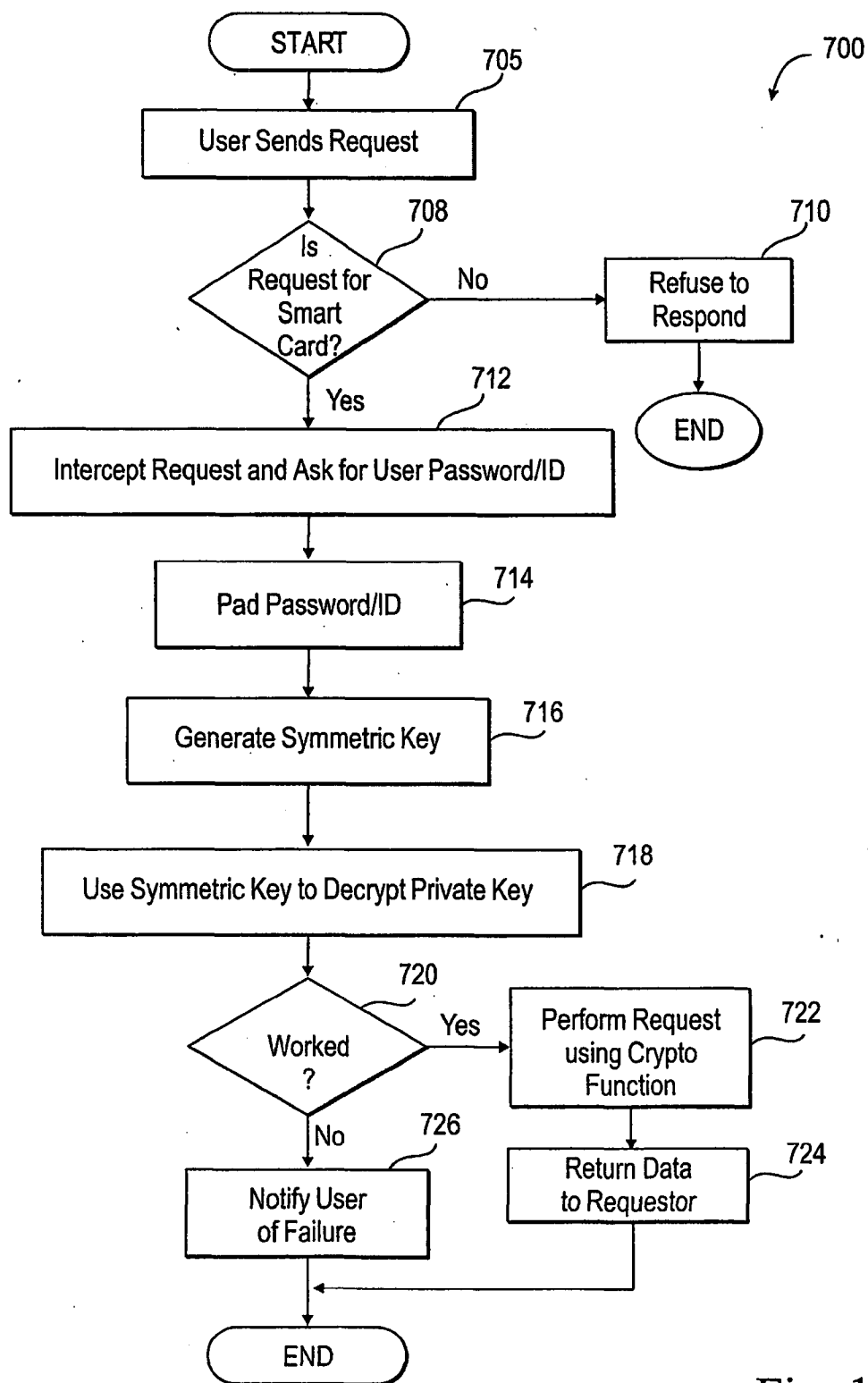


Fig. 10